第 42 卷　第 5 期
2022 年 10 月
桂林电子科技大学学报
Journal of Guilin University of Electronic Technology
Vol. 42, No. 5
Oct. 2022

Ceph 存储系统中节点的容错选择算法

夏亚楠，王　勇

（桂林电子科技大学 计算机与信息安全学院，广西 桂林　541004）

摘　要：Ceph 分布式系统中的数据分布算法仅将容量作为选择存储节点的标准，并未考虑存储节点的网络状态和节点负载。在副本模式下，当三副本中有存储节点并需要修复时，过高的节点负载或者网络负载会导致较大的节点修复时延。针对这个问题，给出了一种基于 Ceph 的节点容错选择（FTNSC）算法。首先利用软件定义网络技术获得实时的网络状态和节点负载信息，作为节点选择方法的数据支撑；然后建立综合考虑节点负载信息的多属性决策数学模型来确定主存储节点位置；最后通过人工蜂群算法根据与主存储节点之间的网络状态和节点性能得到最优次存储节点。实验结果表明，与现有的 CRUSH 算法相比，该算法在提高数据存储节点性能的同时，将失效数据的修复时延减少 2%～29.7%。

关键词：Ceph；软件定义网络；多属性决策；人工蜂群算法；副本放置；容错

中图分类号：TP391　　**文献标志码**：A　　**文章编号**：1673-808X(2022)05-0384-07

A fault tolerant nodes selection algorithm in Ceph storage system

XIA Yanan, WANG Yong

(School of Computer Science and Information Security, Guilin University of Electronic Technology, Guilin 541004, China)

Abstract: The data distribution algorithm in Ceph distributed system only takes the capacity as the standard for selecting storage nodes, and does not consider the network state and node load of storage nodes. In the replica mode, when a storage node in the three replicas needs to be repaired, too high node load or network load will lead to a large node repair delay. To solve this problem, a fault-tolerant node selection algorithm based on Ceph (FTNSC) is proposed. Firstly, the software defined network technology is used to obtain the real-time network state and node load information as the data support of the node selection method; Then, a multi-attribute decision-making mathematical model considering the node load information is established to determine the location of the primary storage node; Finally, the artificial bee colony algorithm is used to obtain the optimal secondary storage node according to the network state and node performance with the primary storage node. Experimental results show that compared with the existing methods, the proposed method can improve the performance of data storage nodes and reduce the data repair delay when nodes fail.

Key words: Ceph; SDN; multiple attribute decision; artificial bee colony algorithm; replica placement; fault tolerant

随着信息化的普及，各行各业每天产生的数据量都在以指数级的速度快速增长，预计到 2025 年，全球数据量相较 2016 年将增加 10 倍[1]。传统的存储模式已经无法应对海量数据的存储需求，而分布式存储系统通过廉价的商用硬件较好地解决了这一问题，目前投入商业使用的有 OpenStack Swift、Amazon EBS、Ceph 等[2]。其中，Ceph 因具有可扩展性、高性能、统一存储以及适用范围广的优势，被广泛应用于当前主流的软硬平台。

作为分布式对象存储典型代表，Ceph 最初由 Weil 等开发提出，十多年间已经有超过 100 家公司（机构）研究与使用 Ceph，其中包括欧洲原子能研究组织、Yahoo、阿里巴巴等。同时由于不同的应用场景对存储系统的关注点不同，吸引许多学者开展了一系列针对 Ceph 性能优化的研究，主要包括读写性能优化、节点工作负载优化、存储数据分布的优化等方

收稿日期：2022-04-18
基金项目：国家自然科学基金(61861013)
通信作者：王勇(1964－)，男，教授，博士，研究方向为云计算、网络信息安全、大数据、人工智能。E-mail：ywang@guet.edu.cn
引文格式：夏亚楠，王勇. Ceph 存储系统中节点的容错选择算法[J]. 桂林电子科技大学学报，2022，42(5)：384-390.

面。为了提高 Ceph 中数据的读写性能，Zhan 等[3]针对 librados 库方法，应用 2 种多线程算法来优化读写性能，使得大文件的下载速度和小文件的上传、下载速度得到了提升；Ceph 节点工作负载方面，Sevilla 等[4]设计了可编程元数据存储系统，将策略和迁移机制分离，可以满足用户不同需求时节点负载均衡；在存储节点选择方面，SHA 等[5]根据 Ceph 的架构和 MapReduce 的特性，提出了一种混合整数线性规划方法来选择最优的存储节点，解决了 Ceph 存储系统中应用 MapReduce 算法配合 CRUSH 算法进行节点选择时系统性能较差的问题。但是这些对 Ceph 集群数据存储位置选择策略和工作负载的研究并未考虑节点间的网络状态信息。

SDN 作为一种新的网络模型，解决了传统的网络状态测量方法存在的网络配置复杂且资源消耗较大的问题[6]。目前 SDN 在网络状态信息测量方面的研究有：Harewood-Gill 等[7]针对现有网络方法无法应对 Qos 要求指数增长的问题，提出利用 SDN 综合考虑时延和带宽的 Q-Routing 算法来选择最佳路径，该算法与 K-Shortest Path 算法相比具有更快的路径计算速度；Okwuibe 等[8]在边缘云环境下提出一种基于 SDN 的资源管理模型，该模型对内存、带宽等资源进行集中管理，并自动计算不同条件下的最佳资源分配方案，根据预定义的约束动态调整分配的资源，降低了系统的部署成本；王勇等[9]设计出一种 SDN 与 Ceph 结合的架构，利用 SDN 获取的信息选择性能最优 OSD 的方式，显著提升了 Ceph 的读性能。这些工作为在 Ceph 中获取网络状态信息来进行节点选择提供了有力依据。

虽然国内外学者在 Ceph 性能优化和利用 SDN 优化网络存储方面取得了很多成果，但较少有研究从利用 SDN 获取的网络信息来增强系统容错性的角度优化 Ceph 性能[10]。因此，考虑 Ceph 集群中节点的负载状况，结合 SDN 中集中控制思想，给出一种提升集群故障节点修复性能的节点容错选择（fault-tolerant node selection based on Ceph，简称 FTNSC）算法。

1 Ceph 数据修复过程及问题描述

Ceph 的三副本模式提供了良好的容错性，使得在出现磁盘故障、服务器宕机等故障时不会出现数据丢失，但在故障发生后仍需对丢失的副本数据进行修复，以保证数据的高可靠性[11]。按照是否能够依靠日志进行修复，Ceph 中存在 2 种修复方式：Recovery 和 Backfill[12]。考虑受损副本能够通过日志进行修复的 Recovery 过程。根据受损副本存在的位置不同，Recovery 有 2 种修复的方式，当降级对象在 Primary PG 上时，Primary PG 所在主 OSD 会通过 Pull 方式从其他的副本拉取待修复对象的权威版本至本地完成修复；当其他副本存在降级对象时，Primary PG 所在主 OSD 会通过 Push 方式主动将待修复对象的权威版本推送到目标副本，由副本完成修复[13]。最后当数据修复工作完成后，副本中的数据恢复一致。若数据放置节点的负载较高或节点间网络状态较差，将会影响数据修复过程的修复时延。

Ceph 中数据的映射过程如图 1 所示，从图 1 可知，Ceph 将数据映射到 OSD 上主要分为 3 步：1）将文件切分为同等大小的多个对象，每个对象获得唯一的 ID，记为 oid。2）利用哈希函数对每个 oid 进行哈希运算，得到的结果与 mask（mask＝PGs－1）值按位进行与操作，生成 PG 的编号 pgid。3）通过 crush 算法将 PG 映射到 OSD 中。此过程运算如式(1)所示。

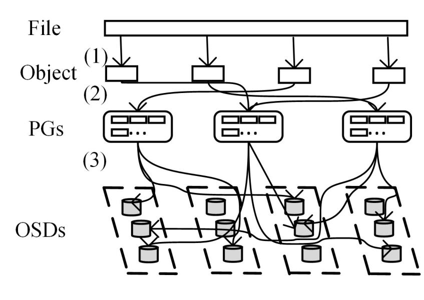


图 1 数据映射过程

$$\text{CRUSH}(\text{pgid},\text{CRUSH_Map},\text{CRUSH_Rule}) = (\text{OSD}_1,\text{OSD}_2,\cdots,\text{OSD}_n), \quad (1)$$

其中：CRUSH_Map 表示保存集群拓扑状态等信息的映射表；CRUSH_Rule 表示副本的容错规则，最终得到存放三副本的 3 个 OSD。其中编号最小的为 Primary OSD（主 OSD），剩余的为 Replica OSD（次 OSD）。在 CRUSH 算法中，CRUSH Maps 作为记录存储集群的层级结构与副本映射以及节点权重的参数，将存储容量作为权重决策的唯一因素，却未考虑节点负载和网络状态对于集群的影响，这会导致节点负载过高或网络性能很差的节点仍会被选为存储节点，影响集群的整体性能。

根据对 Ceph 发生故障时的数据修复过程和节点的选择算法分析发现，虽然 Ceph 的节点选择策略能够将数据均匀地分布在集群中的同时，也保证了集群的稳定性和灵活性，但是由于 CRUSH 过程将节

点的剩余存储容量作为数据映射过程中选择主次 OSD 的唯一决定因素，当有节点发生故障时较高的节点负载和副本之间较差的网络状态会导致节点的修复时延较长，后续故障会导致更高的数据丢失概率。针对上述问题，综合考虑异构节点的性能和副本之间的网络状态，建立优化 Ceph 系统性能的存储节点选择模型，以期在节点发生故障时能够尽快地完成数据修复，减少发生数据损坏的风险。

2 Ceph 分布式存储系统的节点容错选择算法

在具有 N 个 OSD 节点的 Ceph 集群中，每个节点最多可存放一份数据副本，将副本数设为 3，则 Ceph 中的节点容错选择问题可总结为选择合适的放置数据副本的 OSD 集合，使得节点出现故障时，最小化节点的数据修复时延。根据对节点容错选择方法的定义，按两阶段分别对主、次 OSD 进行选择。

2.1 副本放置问题建模

将节点故障后的修复时延定义为 2 部分：节点处理时延和数据传输时延，具体模型构建过程如图 2 所示。

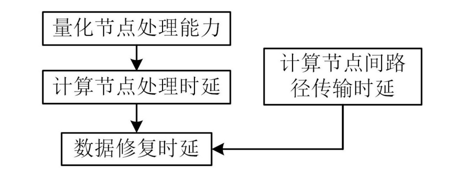


图 2 模型构建流程

对于节点的处理时延，考虑节点异构对于节点数据处理能力的影响，采用齐凤林等[14]对节点处理能力的定义。考虑影响节点处理能力的因素有 CPU 性能、IO 速率、内存及芯片，分别用 f_1, f_2, f_3, f_4 来表示，并为这些影响因素分配相应的权重 w_1, w_2, w_3, w_4，则节点 i 的处理能力可表示为

$$C_i = \sum_{p=1}^{4} w_p f_p 。 \tag{2}$$

假设节点 N_i 需要处理的数据量为 D，能力转化系数为 ∂，则节点 N_i 对数据的处理时延为

$$T_{N_i} = \partial \frac{D}{C_i}。 \tag{3}$$

对于数据的传输时延，参考秦华等[15]对传输时延的定义，将传输路径上各段链路的传输时延之和作为端到端的传输时延，设 $E(N_i, N_j)$ 为节点 N_i 与 N_j 之间的传输路径，b 为路径带宽，则节点 N_i 与 N_j 之间的传输时延为

$$T_{\text{trans}} = \sum_{e \in E(N_i, N_j)} \frac{D}{b_e} 。 \tag{4}$$

在对失效节点的数据进行修复时，若是主 OSD 故障，则需要从次 OSD 拉取需要修复的数据到本地；若次 OSD 发生故障，则需主 OSD 主动将数据 Push 到次 OSD，整个数据修复时间包括源节点和目的节点的处理时间以及数据的传输时间：

$$T_{ij} = \partial \frac{D}{C_i} + \sum_{e \in E(N_i, N_j)} \frac{D}{b_{ij}} + \partial \frac{D}{C_j}。 \tag{5}$$

OSD 节点数量为 N 的集群中，节点故障修复的时间消耗的目标函数可表述为整数线性规划问题：

$$\min \sum_{i \in N} \sum_{j \in N} T_{ij} \varphi_{ij} x_{ij}, \tag{6}$$

$$\text{s. t.} \quad \sum_{i=1}^{N} x_{ij} D_{ij} \leqslant r_j 。 \tag{7}$$

$$\sum_{j=1}^{N} x_{ij} = 2, \ i \in \{1, 2, \cdots, N\}, \tag{8}$$

$$\sum_{i \in N} \sum_{j \in N} \varphi_{ij} = 1, \tag{9}$$

$$\varphi_{ij} \in \{0, 1\}, \forall i \in N, \forall j \in N, \tag{10}$$

$$x_{ij} \in \{0, 1\}, \forall i \in N, \forall j \in N。 \tag{11}$$

其中，约束条件(7)中 r_j 表示节点 j 的剩余容量，D_{ij} 表示从节点 i 传输到节点 j 的数据量，式(7)确保放置副本内容资源请求不超过节点的剩余容量。约束条件(8)表示共有 2 个节点为节点 i 的数据提供副本访问，即保证每份数据的三副本存储。式(9)表示只考虑单点故障即一次修复操作只修复一个失效节点。式(10)中，φ_{ij} 为二进制决策变量，当 φ_{ij} 值为 1 时，表示节点 i 将修复数据发往节点 j，对节点 j 进行故障修复，当 φ_{ij} 值为 0 时则反之。式(11)中，x_{ij} 表示副本放置的决策，$x_{ij} = 1$ 时表示节点 j 为节点 i 提供副本访问，当 x_{ij} 值为 0 时，表示节点 j 为节点 i 提供副本访问。

2.2 基于多属性决策的主 OSD 选择

根据 Ceph 数据映射过程可知，在 Ceph 的节点选择中仅仅以节点的剩余存储容量作为节点被选择的权重因子。对此在考虑节点剩余容量的基础上增加节点 CPU、IO、芯片 3 个指标作为节点选择的权重因素，使用基于理想解的副本选择方法确定出具有最优处理性能的节点，作为副本数据存放的主 OSD。

逼近理想解排序法(technique for order preference by similarity to ideal solution，简称 TOPSIS)是一种有效的多属性决策方案，该方法从归一化的原始数据矩阵中构造出决策问题的正理想解和负理想解，通过计算各方案与正、负理想解的距离作为评价

方案的准则[16]。主 OSD 选择算法设计如下：

步骤 1：决策矩阵构造及其归一化

设基于多属性决策的主 OSD 问题中有 m 个节点，得到节点方案集 $P=\{P_1,P_2,\cdots,P_m\}$，每个节点有 4 个属性指标，构成节点属性集 $A=\{A_1,A_2,A_3,A_4\}$，决策矩阵构造如下：

$$\boldsymbol{T}_{m\times n}=\begin{bmatrix} t_{11} & t_{12} & t_{13} & t_{14} \\ t_{21} & t_{22} & t_{23} & t_{24} \\ \vdots & \vdots & \vdots & \vdots \\ t_{m1} & t_{m2} & t_{m3} & t_{m4} \end{bmatrix} \tag{12}$$

对式(12)所示决策矩阵进行归一化：

$$t'_{ij}=\frac{t_{ij}}{\sqrt{\sum_{i=1}^{m} t_{ij}^{2}}},\ i=1,2,\cdots,m;\ j=1,2,3,4。 \tag{13}$$

步骤 2：构造加权决策矩阵

OSD 节点的 CPU、I/O、内存及芯片对节点处理能力的影响是不同的，对于相对重要的属性如 I/O 分配较大的权重因子，每个属性分配的具体权重 W_j 通过实验的方式获取[17]。得到加权决策矩阵 $\boldsymbol{Z}=(z_{ij})_{m\times n}$，$i=1,2,\cdots,m$；$j=1,2,3,4$

$$z_{ij}=\frac{t_{ij}}{\sqrt{\sum_{i=1}^{m} t_{ij}^{2}}}W_j。 \tag{14}$$

步骤 3：确定正负理想解

$$Z^{+}=(Z_1^{+},Z_2^{+},Z_3^{+})=\max_{i}\{Z_{ij}\mid j=1,2,3,4\}, \tag{15}$$

$$Z^{-}=(Z_1^{-},Z_2^{-},Z_3^{-})=\min_{i}\{Z_{ij}\mid j=1,2,3,4\}, \tag{16}$$

其中，Z^{+} 表示加权决策矩阵的正理想解，由集群中所有候选节点上每种属性的最大值构成；Z^{-} 表示加权决策矩阵的负理想解，由集群中所有候选节点上每种属性的最小值构成。

步骤 4：计算每个候选节点到正负理想解的距离 D_i^{+}、D_i^{-}。

$$D_i^{+}=\sqrt{\sum_{j=1}^{4}(Z_{ij}-Z_j^{+})^2}, \tag{17}$$

$$D_i^{-}=\sqrt{\sum_{j=1}^{4}(Z_{ij}-Z_i^{-})^2}。 \tag{18}$$

步骤 5：计算每个候选节点到最优节点的相对贴近度。

$$C_i^{+}=\frac{D_i^{-}}{D_i^{+}+D_i^{-}},\ i=1,2,\cdots,m, \tag{19}$$

对每个候选节点的相对贴近度 C_i^{+} 进行排序，C_i^{+} 值越大，表示该 OSD 节点的性能越好。在 CRUSH 算法进行 OSD 选择的过程中考虑异构节点的 CPU、I/O、内存和芯片，选择出节点性能最优的节点作为存放副本的主 OSD。

2.3 基于人工蜂群算法的次 OSD 选择

FTNSC 算法在 2.2 节得到的节点性能最优的主 OSD 基础上，考虑节点性能以及与主 OSD 之间的网络状态对数据修复时延的影响，计算出不同候选节点的适应度值，通过人工蜂群算法选择出合适的次 OSD 节点。

人工蜂群(artificial bee colony，简称 ABC)算法是一种模拟蜂群寻找蜜源过程的群体智能优化算法，具有结构简单、控制参数较少及鲁棒性强等特点。人工蜂群中的蜜蜂分为 3 种：雇佣蜂、跟随蜂和侦查蜂。3 种蜜蜂通过分工协作开采蜜源，并通过蜜源的标记与不断分享更新位置寻找最优蜜源，即问题的最优解[18]。其中，蜜源的位置对应优化问题的一个可行解，蜜源的蜜量对应适应度值。算法步骤如下：

1)解的构造和编码：基本的 ABC 算法适用于连续型论域，并不适用于本节点选择问题的整数编码方式，因此将其进行离散化处理。假设集群中共有 N 个 OSD 节点，主 OSD 的数据需要选择 2 个次 OSD 进行放置，则解的构造如式(20)所示。

$$x_i=\{x_i^1,x_i^2\},\ x_i^j\in[x_{\min}^j,x_{\max}^j],\ i=1,2,\cdots,N。 \tag{20}$$

其中：x_i 为第 i 个蜜源，即第 i 个节点选择方案；x_i^j 表示第 j 个次 OSD 选择为编号为 i 的节点存储数据；$x_{\max}^j$、$x_{\min}^j$ 分别代表集群中 OSD 节点编号的上限和下限。如编码{3，17}，表示第一个次 OSD 节点选择集群中 id=3 的节点，第二个次 OSD 选择集群中 id=3 的节点。

2)蜜源初始化阶段：设有 S_N 个蜜源 $\{x_1,x_2,\cdots,x_{S_N}\}$，雇佣蜂数量与蜜源数量相等，每个蜜源 $x_i=\{x_i^1,x_i^2,\cdots,x_i^{D_n}\}$ 均为一个 D_n 维向量，代表一个初始解，D_n 设为 2，$i\in\{1,2,\cdots,S_N\}$，则初始蜜源为

$$x_i^j=\mathrm{round}(x_i^j+r_{ij}(x_{\max}^j-x_{\min}^j))。 \tag{21}$$

其中：$r_{ij}\in(0,1)$ 表示(0，1)内均匀分布的随机数，$j\in\{1,2,\cdots,D_n\}$，$x_{\max}^j$、$x_{\min}^j$ 代表蜜源初始解的上下限。FTNSC 算法采取整数编码方式，在基本的 ABC 算法上进行改进，使用 round()函数将产生的蜜源值取整。若式(21)产生的初始蜜源不能满足节点选择的限制条件(7)～(11)，则淘汰此解重新生成新解。

3)确定适应度函数值：对于每个初始解，根据式(22)计算其适应度值，判断蜜源的优劣。

$$\mathrm{fit}(x_i)=\begin{cases}\dfrac{1}{1+\mathrm{fit}(x_i)},\ \mathrm{fit}(x_i)\geqslant 0;\\ 1+\mathrm{abs}(\mathrm{fit}(x_i)),\ \mathrm{fit}(x_i)<0。\end{cases} \tag{22}$$

设选择出来的最佳主 OSD 节点编号为 b_m，则基于人工蜂群算法的次 OSD 选择算法的适应度函数表示选取具有节点处理性能以及与节点 b_m 之间网络状态最好的节点作为次 OSD，来减少发生故障时数据的修复时延。

$$\mathrm{fit}(x_i)=\sum_{e\in E(N_i,N_j)}\frac{D}{b_{b_m j}}+\partial\frac{D}{C_j}。 \tag{23}$$

4）雇佣蜂阶段：雇佣蜂在蜜源的邻域内进行搜索，基本 ABC 算法根据式(24)产生新的蜜源。

$$v_i^j=\mathrm{round}(x_i^j+r_{ij}(x_i^j-x_k^j))。 \tag{24}$$

其中，$k\in\{1,2,\cdots,S_N\}$，$k\neq i$ 即 x_k^j 是不同于 x_i^j 的蜜源。通过 round()函数对新蜜源取整，若新蜜源不是满足节点选择限制条件的可行解，则根据式(24)重新生成新蜜源。雇佣蜂根据贪婪原则进行蜜源选择，若新蜜源的适应度值高于原蜜源，则用新蜜源替换原蜜源；否则，保持原蜜源不变，并将此蜜源的开采轮数加 1。

4）跟随蜂阶段：雇佣蜂完成邻域搜索后，跟随蜂接收到雇佣蜂分享的信息后进行进一步开采，通过轮盘赌算法按式(25)计算的概率选择蜜源，蜜源的适应度值越大，被跟随蜂选择的概率越大。

$$P_i=\frac{\mathrm{fit}(x_i)}{\sum_{i=1}^{S_N}\mathrm{fit}(x_i)}。 \tag{25}$$

5）侦查蜂阶段：设定蜜源的开采上限为 l 次，若蜜源经过 l 次迭代后适应度值仍无变化，则为了防止陷入局部最优，将此蜜源淘汰，用式(21)产生的新蜜源替换。

迭代完成后，将适应度值最大的蜜源作为最优解，得到存放副本的 2 个次 OSD 节点，算法结束。

3 实验设置与性能评估

为评估 FTNSC 算法的性能，在 Mininet[19] 网络模拟器上模拟和构建 Ceph 集群拓扑，并在 SDN 控制器 Ryu[20] 上部署 FTNSC 算法。仿真实验环境为：Intel Core i5-4210，操作系统为 VMware Ubuntu 14.0，4 GiB 内存，Mininet 版本为 2.3.0，Ryu 4.34 以及 1.3 版本的 Openflow 协议。

将拓扑中存储节点的个数设为 24，节点中的数据采取三副本法则进行设置，网络拓扑采用无向完全图。模拟实验场景中的异构存储节点的处理能力服从文献[8]，各异构节点性能参数及其取值范围为{CPU，[1，90]}，{I/O，[21，265]}，{内存，[0.3，76]}，{芯片，[0.5，23]}，对应的权重分别为 30%，40%，20%，10%，节点能力转化系数 ∂ 为 0.3。设置集群网络拓扑中各链路的可用带宽随机均匀分布在 50～100 Mbit/s，默认链路 (V_i, V_j) 与 (V_j, V_i) 相互独立且大小相等。

Ceph 集群中的失效节点的数据修复考虑 2 种情况：首先是降级对象发生在主 OSD 节点，此时主 OSD 通过 Pull 方式从 2 个次 OSD 上拉取缺少的数据；当次 OSD 存在降级对象时，主 OSD 通过 Push 方式主动将缺失数据推送到相应的副本，本实验中随机设定故障节点来模拟 Ceph 集群中单节点失效的修复过程。算法分为 2 个阶段，第一个阶段为主 OSD 的选择过程，通过 SDN 获取异构节点的 CPU、IO、内存以及芯片 4 个指标，并将其作为节点选择的权重因素，使用基于理想解的副本选择方法确定出具有最优处理性能的节点。第二个阶段在第一阶段选择出主 OSD 基础上，考虑节点性能以及其与主 OSD 之间的网络状态对数据修复时延的影响，计算出不同候选节点的适应度值，通过人工蜂群算法选择出合适的次 OSD 节点。算法中涉及的参数如表 1 所示。

表 1 算法参数取值

参数	值
节点的数量	24
可行解维度	2
迭代次数	20
蜜源数量	20
雇佣蜂数量	20
跟随蜂数量	60
蜜源最大丢弃次数	24

将 FTNSC 算法与 Ceph 云存储系统原生的 CRUSH 算法以及 NSMBD 算法[8] 进行对比实验。其中，CRUSH 算法作为 Ceph 云存储系统数据映射过程的经典算法，将存储节点按剩余存储容量进行排序，选择存储容量性能最优的 3 个节点。NSMBD 是一种有效的针对小文件的 Ceph 存储系统节点选择方法，其利用 SDN 获取集群中的网络带宽性能，并用逼近理想解排序法选择出性能最优的 3 个节点来存放数据副本，以提高读写操作的吞吐量和响应时间。实验测试数据修复时延，从所选节点的 CPU、IO、内存、芯片角度对比 3 种算法的性能，每组实验测试 5 次，取平均值作为实验结果。

3.1 节点故障的数据修复时延

为了测试 3 种不同算法对于不同大小数据修复对

象的修复时延的性能,如图 3 所示,实验中制定了 50、100、150、200、250、300 MB 6 种对象的工作量,比较 Ceph 原有的 CRUSH 算法、NSMBD 算法和本算法在不同修复数据对象时,完成故障节点数据修复的时间。

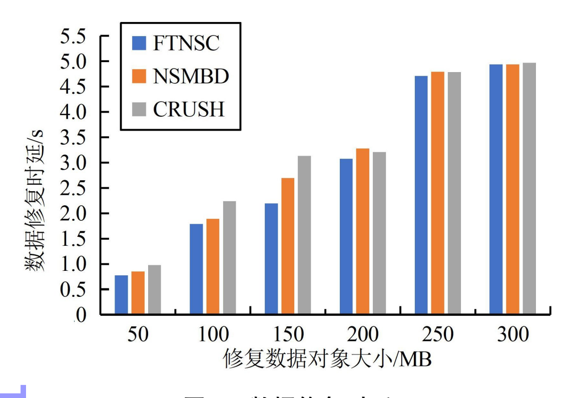


图 3 数据修复时延

从图 3 可看出,对于 6 种不同规模的修复数据,FTNSC 算法均拥有最优修复时延,且对于 200 MB 以下的小规模修复数据有明显提升,当修复数据的规模大于 200 MB 时,数据修复时延的提升较小。由于 CRUSH 算法选择集群中剩余容量性能较优的节点存储数据,但在节点发生失效而进行数据修复时,仍无法避免较差的网络性能和节点其他性能的影响,导致修复时延较大。随着修复数据量增大,上述缺陷得到一定程度的改善,但 CRUSH 算法的修复时延在 3 种算法中仍最长。在不同的修复数据规模场景下,FTNSC 算法的数据修复时延较 CRUSH 算法减少了 2%~29.7%,较 NSMBD 算法减少了 1.8%~18.4%,这是由于在实验场景中引用了存储节点处理能力的异构和拓扑网络状况,为副本数据选择性能最优的节点进行存储。

3.2 三副本节点性能

图 4 为在修复数据大小为 150 MB,异构节点的能力转化系数为 0.3 时,FTNSC 算法、NSMBD 算法和 CRUSH 算法所选 OSD 节点的 CPU、IO、内存及芯片性能情况。

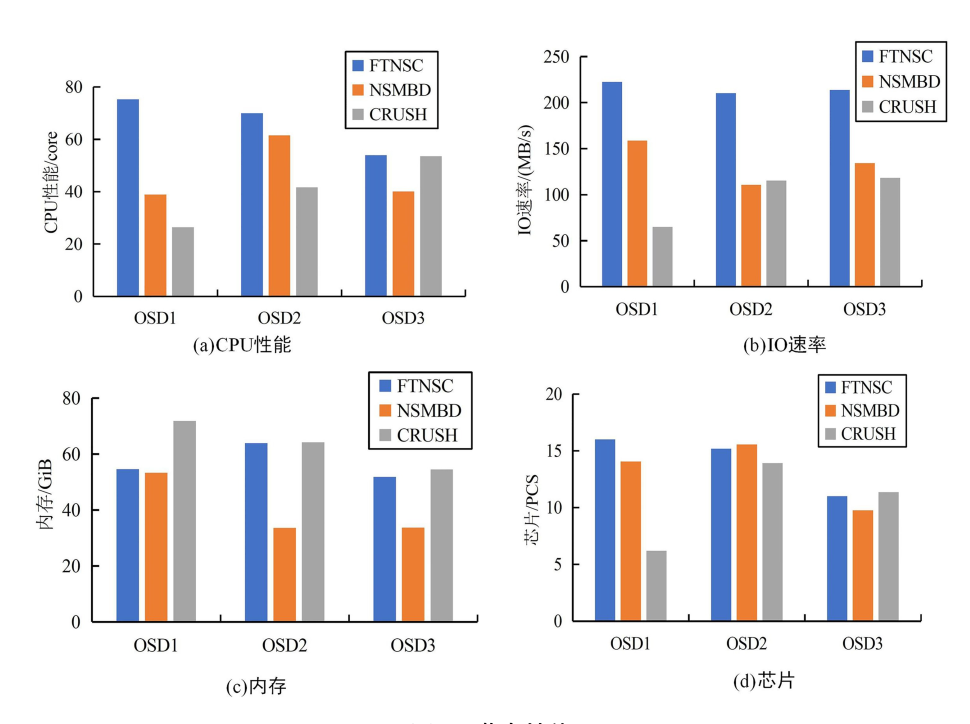


图 4 节点性能

在节点的 CPU、IO、芯片性能中,FTNSC 算法所选主 OSD 的性能明显优于 CRUSH 算法和 NSMBD 算法,这是由于 FTNSC 算法不仅考虑了集群节点的 4 种资源指标,还分别考虑了各个资源指标的权重情况,降低了所选节点某项资源指标性能较差的可能。对于次 OSD,FTNSC 算法在 CPU 性能和 IO 速率上表现较好,但在内存和芯片性能上提升并不明显。Ceph 存储系统中的 2 种数据修复都依靠主 OSD 通过 Pull 或 Push 方式从次 OSD 中选取一个节点获取或发送失效数据,因此相对次 OSD 而言,主 OSD 在失效节点修复过程中具有更加重要的作用。在内存方面,CRUSH 算法性能优于 FTNSC 算法,这是由于 CRUSH 算法以内存作为节点选择的唯一标准。综合 4 种指标,FTNSC 算法所选节点的性能优于另外 2 种算法,使得在发生节点失效时,节点具有更高的处理能力,从而减少数据修复的时延。

4 结束语

针对云存储系统中大量使用廉价商用硬件引起的频繁的节点失效问题,提出了一种 Ceph 存储系统中 FTNSC 算法,通过优化副本放置的节点位置来减少节点失效时的数据修复时延。首先利用软件定义网络技术,获得实时的网络状态和节点负载信息,并将其作为节点选择方法的数据支撑;然后通过建立综合考虑节点负载信息的多属性决策数学模型确定主存储节点位置;最后通过人工蜂群算法,根据与主存储节点之间的网络状态以及节点性能得到最优次存储节点。实验结果表明,FTNSC 可以提高所选节点性能,同时减少数据的修复时延,降低集群中数据丢失的风险,从而提高整个存储系统的可靠性。下一阶段将考虑将 FTNSC 算法应用到开源 Ceph 云存储平台上,产生实际价值。同时可以在 SDN 监控的消耗问题上做进一步的研究,讨论测量过程中对 Ryu 控制器的性能和交换机的带宽资源的消耗。

参考文献:

[1] NIELSEN L, VESTERGAARD R, YAZDANI N, et al. Alexandria: a proof-of-concept implementation and evaluation of generalised data deduplication[C]//2019 IEEE Globecom Workshops. Piscataway, NJ: IEEE Press, 2019: 1-6.

[2] HUA Y S, SHI X H, HE K, et al. Loomio: object-level coordination in distributed file systems[J]. IEEE Transactions on Parallel and Distributed Systems, 2021, 33(8): 1799-1810.

[3] ZHAN K, PIAO A H. Optimization of ceph reads/writes based on multi-threaded algorithms[C]//2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems. Piscataway, NJ: IEEE Press, 2016: 719-725.

[4] SEVILLA M A, WATKINS N, MALTZAHN C, et al. Mantle: a programmable metadata load balancer for the ceph file system[C]//Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. Piscataway, NJ: IEEE Press, 2015: 1-12.

[5] SHA E H M, LIANG Y, JIANG W, et al. Optimizing data placement of mapreduce on ceph-based framework under load-balancing constraint[C]//2016 IEEE 22nd International Conference on Parallel and Distributed Systems . Piscataway, NJ: IEEE Press, 2016: 585-592.

[6] 周桐庆,蔡志平,夏竟,等.基于软件定义网络的流量工程[J].软件学报,2016,27(2):394-417.

[7] HAREWOOD-GILL D, MARTIN T, NEJABATI R. The performance of Q-learning within SDN controlled static and dynamic mesh networks[C]//2020 6th IEEE Conference on Network Softwarization. Piscataway, NJ: IEEE Press, 2020: 185-189.

[8] OKWUIBE J, HAAVISTO J, KOVACEVIC I, et al. SDN-enabled resource orchestration for industrial iot in collaborative edge-cloudnetworks[J]. IEEE Access, 2021, 9: 115839-115854.

[9] 王勇,叶苗,何倩,等.基于软件定义网络和多属性决策的 Ceph 存储系统节点选择方法[J].计算机学报,2019,42(2):93-108.

[10] ALRAZIB M, JAVEED D, KHAN M T, et al. Cyber threats detection in smart environments using SDN-enabled DNN-LSTM hybrid framework[J]. IEEE Access, 2022.

[11] WEIL S A, LEUNG A W, BRANDT S A, et al. Rados: a scalable, reliable storage service for petabyte-scale storage clusters[C]//Proceedings of the 2nd International Workshop on Petascale Data Storage: Held in Conjunction with Supercomputing'07. New York, NY: ACM Press, 2007: 35-44.

[12] STER D C, LAMANNA M, MASCETTI L, et al. Ceph-based storage services for run2 and beyond[C]//Journal of Physics: Conference Series. IOP Publishing, 2015, 664(4): 042054.

[13] 潘松杜.多数据中心间异地存储管理平台的设计与实现[D].南京:东南大学,2016:27-28.

[14] 齐凤林,宫庆媛,周扬帆,等.分布式存储再生码数据修复的节点选择方案[J].计算机研究与发展,2015,52(增刊 2):68-74.

[15] 秦华,阎钢.基于 OpenFlow 网络的数据中心服务器负载均衡策略[J].计算机工程,2016,42(3):130-137.

[16] 柯文龙,王勇,叶苗,等.Ceph 云存储网络中一种业务优先级区分的多播流调度方法[J].通信学报,2020,41(11):40-51.

[17] LUO Y Q, XIA J B, CHEN T. Comparison of objective weight determination methods in network performance evaluation[J]. Journal of Computer Applications, 2009, 29(10): 2624-262.

[18] HAN A Y, ZHOU S Q, JIAN X H, et al. Short-term load forecasting model based on RBF neural network optimized by artificial bee colony algorithm[C]//2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering Piscataway, NJ: IEEE Press, 2021: 486-489.

[19] LANTZ B, O'CONNOR B. A mininet-based virtual testbed for distributed SDN development[J]. ACM SIGCOMM Computer Communication Review, 2015, 45(4): 365-366.

[20] ISLAM M T, ISLAM N, ALREFAT M. Node to node performance evaluation through ryu SDN controller[J]. Wireless Personal Communications, 2020, 112(1): 555-570.

编辑:梁王欢

第 42 卷　第 5 期
2022 年 10 月

桂林电子科技大学学报
Journal of Guilin University of Electronic Technology

Vol. 42, No. 5
Oct. 2022

基于图神经网络的子图匹配符号算法

杨　欣，徐周波，陈浦青，刘华东

（桂林电子科技大学 计算机与信息安全学院，广西 桂林　541004）

摘　要：子图匹配是图数据分析中的基础问题，具有重要的研究意义。针对子图匹配求解算法存在大量冗余搜索的问题，提出了一种基于图神经网络的子图匹配符号算法。该算法利用图神经网络技术聚合节点的邻域信息，得到包含图局部属性和结构的特征向量，以该向量作为过滤条件得到查询图的节点候选集 C。此外，优化匹配顺序并利用符号 ADD 操作在数据图中构建 C 的各个候选区域，减少了子图枚举验证过程中的冗余搜索。实验结果表明，与 VF3 算法相比，该算法有效地提高了子图匹配的求解效率。

关键词：子图同构；图匹配问题；图神经网络；代数决策图；候选区

中图分类号：TP391.4　　　**文献标志码**：A　　　**文章编号**：1673-808X(2022)05-0391-07

Subgraph matching symbol algorithm based on graph neural network

YANG Xin, XU Zhoubo, CHEN Puqing, LIU Huadong

(School of Computer Science and Engineering, Guilin University of Electronic Technology, Guilin 541004, China)

Abstract: Subgraph matching is a fundamental problem in graph data analysis and has important research significance. Aiming at the problem of a large number of redundant searches in the subgraph matching algorithm, a subgraph matching symbol algorithm based on graph neural network(SSMGNN) was proposed. The algorithm used the graph neural network technology to aggregate the neighborhood information of nodes, and obtained the feature vector containing the local attributes and structure of the graph, and used the vector as the filter condition to obtain the node candidate set C of the query graph. In addition, optimizing the matching order and using symbolic ADD operations to construct each candidate region of C in the data graph reduced redundant searches during subgraph enumeration verification. The experimental results show that, compared with the VF3 algorithm, the algorithm effectively improve the solving efficiency of subgraph matching.

Key words: subgraph isomorphism; graph matching problem; graph neural network; ADD; candidate region

图作为一种数据结构，可以有效刻画事物之间的关系，现实世界中的许多复杂问题都可以用图进行抽象表示。子图匹配问题作为图分析中最基本的问题之一，其目标是在数据图 g 中查找与查询图 q 同构的所有子图[1]，在图像检索、化学分子式检索、知识图谱查询和社交网络分析等领域有着广泛应用[2]。由于子图匹配问题属于 NP 难问题，随着数据规模的急剧增加，其求解的复杂度呈指数增长，因此广大研究者致力于扩大子图匹配问题的求解规模和提升求解效率。

在过去几十年，大量子图匹配算法被提出，其中基于回溯搜索的算法[3-13]更是被广泛研究。著名的 Ullmann 算法[3]于 1979 年提出，该算法基于回溯的树搜索在数据图 g 中枚举出与查询图 q 匹配的所有子图。由于 Ullmann 算法只采用了简单的剪枝策略，无法高效地减少搜索空间。VF2[4]算法在 Ullmann 算法的基础上，将节点的邻居信息作为约束条件，增强了剪枝效果，有效地减少了搜索空间。GraphQL 算法[5]提出以查询图的深度优先搜索生成树过滤数据图中的节点。Spath 算法[6]采用对查询图进行层次遍历并以每层节点的信息作为约束条件，引入复杂的结构与语义信息过滤数据图中的节点。

收稿日期：2022-04-15

基金项目：国家自然科学基金(61762027)；广西自然科学基金(2017GXNSFAA198172)

通信作者：徐周波(1976－)，女，教授，博士，研究方向为符号计算、智能规划、约束求解。E-mail：xzbli_11@guet.edu.cn

引文格式：杨欣，徐周波，陈浦青，等. 基于图神经网络的子图匹配符号算法[J]. 桂林电子科技大学学报，2022，42(5)：391-397.